

ANP-003 — Execution Ledger Protocol

Status: Draft **Version:** v0.1-alpha **Author:** ArchivoNulo Global **Series:** ANP Canonical Protocols **Created:** 2026-03-08

1. Introduction

1.1 General Context

Autonomous systems are rapidly becoming operational infrastructure across enterprise, institutional, and digital governance environments. However, execution pipelines remain opaque, heavily dependent on post-factum audits, manual compliance procedures, and external consulting layers that introduce cost, delay, and systemic risk.

Current challenges include:

- Limited real-time observability of execution flows
- Forensic-only audit models (reactive instead of preventive)
- High dependency on third-party consulting and external assessors
- Weak traceability in distributed AI workflows
- Regulatory exposure due to unverifiable runtime behavior

These gaps become critical under emerging regulatory frameworks that require explainability, traceability, and runtime accountability for autonomous decision systems.

1.2 Positioning Within the ANP Series

The ANP protocol series defines a progressive integrity stack:

- **ANP-001 — Deterministic State Stasis** Establishes immutable state anchoring to prevent drift and nondeterministic divergence.
- **ANP-002 — Agent Integrity Protocol** Ensures agent-level trust through cryptographic proofs and admissibility enforcement.
- **ANP-003 — Execution Ledger Protocol** Operationalizes runtime integrity by recording and verifying execution flows within an immutable cryptographic ledger.

ANP-003 forms the auditable backbone of the ecosystem, converting theoretical integrity guarantees into verifiable operational evidence.

1.3 Core Objectives

ANP-003 aims to:

1. Enable real-time sovereign audit without human intermediaries

2. Guarantee immutability of execution records
3. Allow deterministic reconstruction of execution flows
4. Eliminate reactive audit dependency
5. Reduce operational compliance costs
6. Provide verifiable evidence for regulators and partners

1.4 Scope and Limitations

In Scope

- Runtime execution of autonomous agents
- AI-driven workflows
- Distributed digital supply chains
- Multi-agent orchestration environments

Out of Scope

- Model training lifecycle
 - Dataset provenance pipelines
 - Hardware-level trusted execution
-

2. Definitions and Core Concepts

2.1 Execution Ledger

A distributed append-only ledger that records execution state transitions as immutable cryptographic entries.

Each entry contains:

- Sequence value
- Previous state hash
- Action metadata
- Integrity proofs
- Cryptographic seal

2.2 Real-Time Audit

Continuous verification performed at commit time through deterministic re-derivation of states.

No mutable timestamps are required; ordering is guaranteed via sequence values and hash chaining.

2.3 Sovereign Audit Model

The ledger enables independent verification without privileged access.

Any regulator, partner, or auditor can validate execution authenticity using public proofs.

2.4 Terminology Integration

Commit Boundary Inherited from ANP-002. Defines the admissibility checkpoint prior to ledger inclusion.

Deterministic Stasis Inherited from ANP-001. Anchors canonical initial states.

Multi-Tenant Isolation Logical separation of ledgers by jurisdiction, organization, or operational domain.

3. Protocol Architecture

3.1 Core Components

3.1.1 Ledger Core Append-only cryptographic structure responsible for storing execution records.

Supported topologies:

- Linear hash chain
- Directed acyclic graph (DAG)

Each ledger entry includes:

- Sequence value
- State hash
- Agent integrity proof
- Action metadata
- Cryptographic seal

3.1.2 Admissibility Checker Pre-commit enforcement module that validates:

- Structural invariants
- Policy compliance
- Integrity proofs
- Deterministic transitions

3.1.3 Audit Engine Deterministic replay engine for execution reconstruction.

Capabilities:

- State re-derivation
- Consistency verification
- Failure detection

- Proof validation

3.1.4 Integration Layer API connectors for interoperability with:

- Agent frameworks
 - AI orchestration stacks
 - Enterprise pipelines
 - RAG infrastructures
-

3.2 Operational Flow

Stage 1 — Pre-Execution

- Agent submits action
- Admissibility proofs verified

Stage 2 — Commit Time

- Entry created
- Sequence assigned
- State anchored

Stage 3 — Post-Execution

- Outcome recorded immutably
- Integrity sealed

Stage 4 — Sovereign Query

- External verifiers validate proofs
 - Privacy preserved
-

3.3 State Model (S0–S6)

S0 — Canonical Anchoring Initial immutable state registration.

S1 — Compliance Proofs Validation of admissibility requirements.

S2 — Integrity Enforcement Policy and invariant verification.

S3 — Execution Recording Ledger append operation.

S4 — Failure Interdiction Invalid transitions blocked.

S5 — Sovereign Verification Independent proof validation.

S6 — Audit Reconstruction Deterministic replay and confirmation.

4. Technical Specifications

4.1 Implementation Requirements

Languages

- Python / Go — core systems
- Rust — cryptographic components

Dependencies

- Zero-knowledge proof frameworks
- Ledger storage engines
- Cryptographic libraries

4.2 Security Requirements

- Resistance to TOCTOU vulnerabilities
- Admissibility latency: <100 ms
- Append latency: <1 s
- Distributed throughput: 1M+ tx/day

4.4 Compliance Levels

L1 — Hash Integrity Basic immutability checks

L2 — Signature Validation Authorship verification

L3 — Structural Enforcement Invariant validation

L4 — Cryptographic Proofs Advanced integrity assertions

L5 — Full Sovereign ZKP Privacy-preserving verification

5. Cryptographic Specifications

5.1 Hashing

Standard: SHA3-384 **Purpose:** Collision resistance, pre-image security, state anchoring

Entry seal:

$H = \text{SHA3-384}(\text{sequence} || \text{prev_hash} || \text{metadata} || \text{proofs})$

5.2 Digital Signatures

Algorithm: Ed25519 **Purpose:** Authorship verification and non-repudiation

5.3 Zero-Knowledge Proofs

Frameworks: zk-SNARKs Purpose: Proof of compliance without data disclosure

Used for:

- Admissibility validation
- Sovereign audit queries

5.4 Ledger Structure

Entry tuple:

(sequence, prev_hash, payload, proofs, signature)

5.5 Merkle Integration

Batch verification and inclusion proofs supported via Merkle roots.

5.6 Optional Privacy Layer

- AES-256-GCM payload encryption
 - Homomorphic proofs
 - Zero-knowledge rollups
-

6. Audit and Verification Model

6.1 Deterministic Replay

Execution flows reconstructed from genesis state.

6.2 Independent Verification

External entities validate:

- Chain integrity
- Proof correctness
- Signature authenticity
- Deterministic state transitions
- Tamper-evident logs

6.3 Sovereign Queries

Succinct proofs allow property validation without exposing full records.

7. Benefits

7.1 Operational

- 70–90% reduction in manual audits
- Lower compliance overhead
- Faster verification cycles

7.2 Systemic

- Drift prevention
- Execution transparency
- Failure containment

7.3 Institutional

- Regulatory readiness
 - Partner trust
 - Sovereign governance capability
-

8. Use Cases

8.1 Enterprise

Real-time audit for AI workflow pipelines

8.2 Institutional

National digital governance and regulatory compliance

8.3 Deeptech Infrastructure

Resilient autonomous systems and distributed coordination

9. Risks and Mitigations

9.1 Risks

- Ledger scalability pressure
- Cryptographic overhead
- Adversarial admissibility attacks

9.2 Mitigations

- Modular architecture
- Sharded ledger design

- Adversarial stress testing

9.3 Limitations

- Runtime focus only
 - Requires crypto-ready infrastructure
-

10. Security and Threat Model

Mitigations include:

- Collision resistance
- Anti-replay sequencing
- Constant-time cryptography
- Forward migration to post-quantum standards

Formal verification and independent security audits recommended.

11. Conclusion

ANP-003 transforms execution integrity into an operational, verifiable, and sovereign audit layer.

It closes the foundational loop of the ANP stack by converting:

Deterministic State $\rightarrow$ Agent Integrity $\rightarrow$ Execution Verifiability

This protocol establishes the infrastructure required for autonomous systems governance at global scale.

12. Future Extensions

ANP-003 enables:

- Multi-agent shared ledgers
- Cross-ledger proofs
- Distributed governance fabrics

Next protocol:

ANP-004 — Multi-Agent Coordination Protocol

13. Call for Validation

Community review, institutional pilots, and formal verification initiatives are encouraged.

Issues, discussions, and proposals should follow canonical repository governance processes.

Canonical Artifacts

- Canonical Markdown Document
 - Cryptographic Checksum (SHA3-384)
 - Digital Signature
 - Version Metadata
 - Public Archival Record
-

End of Document